

Semi-Supervised Learning with Heterophily

Wolfgang Gatterbauer
Carnegie Mellon University
gatt@cmu.edu

ABSTRACT

We propose a novel linear semi-supervised learning formulation that is derived from a solid probabilistic framework: belief propagation. We show that our formulation generalizes a number of label propagation algorithms described in the literature by allowing them to propagate generalized assumptions about influences between classes of neighboring nodes. We call this formulation *Semi-Supervised Learning with Heterophily* (SSL-H). We also show how the affinity matrix can be learned from observed data with a simple convex optimization framework that is inspired by locally linear embedding. We call this approach *Linear Heterophily Estimation* (LHE). Experiments on synthetic data show that both approaches combined can learn heterophily of a graph with 1M nodes, 10M edges and few labels in under 1min, and give better labeling accuracies than a baseline method in the case of small fraction of explicitly labeled nodes.

1. INTRODUCTION

Graph-based Semi-Supervised Learning (SSL) methods define a graph where the nodes are labeled and unlabeled examples in the dataset, and where (potentially weighted) edges reflect the similarity of examples [26]. Given this input, the goal of SSL is to infer the labels of the unlabeled data. Existing methods commonly assume “smoothness” of labels over the graph, i.e. a certain *homophily* or *assortative mixing* property in the network that results in “birds of a feather flock together.” However, the reverse is often true in actual data and is also called *heterophily* (“opposites attract”). Previous work has looked into ways to adapt the existing SSL frameworks to handle both similarity *and dissimilarity* (e.g., [7] and [21]). These methods are limited to expressing binary dependencies between nodes (e.g., more similar or more dissimilar). We are interested in more general constraints among classes and also how to learn them from data. For example, assume a social dating network with three different kinds of classes amongst its users. Class 1 prefers to date users of class 2 (and v.v.), whereas users

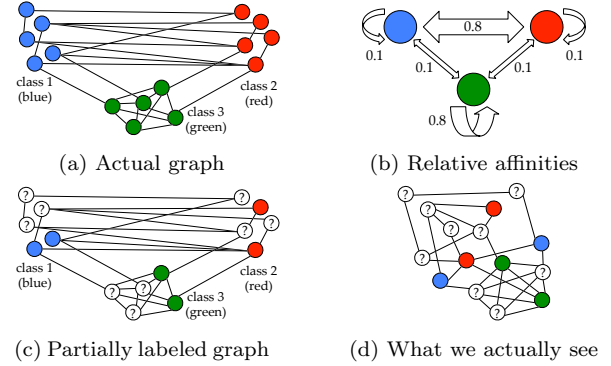


Figure 1: (a,b): Real networks are formed based on relative affinities between the classes of nodes (here represented by their color). (c,d): The problem we are studying in this paper: We are given a *partially* labeled graph (i.e. the adjacency matrix \mathbf{W} and a few classes of nodes) but we do not know the relative affinities between classes. How can we *efficiently* and *simultaneously* learn (i) the relative affinities and (ii) the labels for the unlabeled nodes?

of class 3 prefer to date among themselves (see Fig. 1a).¹ We call these relations simply *heterophily relations*, as they naturally generalize notions of similarity/dissimilarity and assortative or disassortative mixing.²

In this paper, we show how existing SSL methods (e.g., those based on Local and Global Consistency (LGC) [25] or Harmonic Function methods (HF) [27]) can be generalized in a natural way as to allow to propagate heterophily from labeled to unlabeled data. This allows us to propagate label information through a graph in the presence of heterophily, thus, generalizing the commonly implied smoothness assumption between nodes of similar labels. In this regard, this work draws heavily upon and provides a generalization of our recent work on linearizing the update equations of belief propagation [6].

Our key idea relies on well-known and intuitive facts from Markov chains and properties of symmetric doubly stochastic matrices. We illustrate with the help of Fig. 2c: when a stochastic vector \mathbf{x} (which, e.g., represents the inferred label distribution of a certain node) is transformed (or “modulated”) with a doubly stochastic matrix \mathbf{H} , then the re-

¹Notice that we imply undirected, i.e. symmetric relationships throughout this paper.

²In the physics community, homophily or assortative mixing is also referred to as “ferromagnetism,” while heterophily or disassortative mixing is referred to as “antiferromagnetism.”

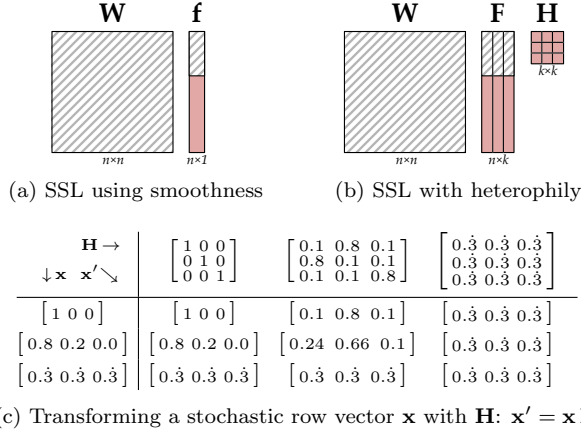


Figure 2: (a) In standard Semi-Supervised Learning (SSL), we learn a labeling function \mathbf{f} for unlabeled data (shown as red part of \mathbf{f}) based on partially labeled data (shown as hatched part of \mathbf{f}) by using the known graph structure \mathbf{W} and various assumptions of smoothness (= homophily). (b) In our generalization, we allow arbitrary assumptions of coupling strengths between classes of neighboring nodes (e.g., “opposites attract”). (c) We express such arbitrary couplings with the help of a simple linear transformation by a symmetric and doubly stochastic affinity matrix \mathbf{H} . In this paper, we show how to simultaneously learn both, the heterophily matrix \mathbf{H} and the missing labels from existing data.

sulting vector \mathbf{x}' is still stochastic.³ This simple fact allows us to express a node’s label distribution based on the label distribution of its neighbors and, furthermore, to express arbitrary similarity or dissimilarity relationships between neighboring nodes. We call such a modulating \mathbf{H} matrix interchangeably affinity, coupling, modulation, or *heterophily matrix*, as it captures the pair-wise preference or coupling strengths between nodes and their classes in a network. This, in turn, allows use to derive a generalized formulation of standard semi-supervised learning approaches and to derive them back by using the identify matrix \mathbf{I}_k as affinity matrix. The original Linear Belief Propagation (LinBP) formulation is one of several possible linear label propagation approaches. We thus call our first contribution Semi-Supervised Learning with Heterophily (SSL-H).

Our second key contribution is a simple way to learn the heterophily matrix \mathbf{H} from the data. Here, we draw an interesting connection to the Locally Linear Embedding (LLE) [15] framework. While the problems in LLE and our setup are different (LLE tries to find an optimal linear embedding across neighbors to reduce the dimensionality of the data; we find an optimal “heterophily explanation” between the classes of neighbors in a network), the mathematical formalism is similar. We thus call our second contribution Linear Heterophily Estimation (LHE). Together, SSL-H and LHE allow us to both (i) learn heterophily and to (ii) label unlabeled data (see Fig. 2b).

Contributions. The two main contributions are thus:

1. *Semi-Supervised Learning with Heterophily (SSL-H)*: We generalize semi-supervised learning to general heterophily assumptions. The commonly used smooth-

ness assumption is the important special case with the identify matrix as affinity matrix.

2. *Linear Heterophily Estimation (LHE)*: We show how heterophily can be learned from existing partially labeled data even at the presence of few labels. This resolves the issue that the propagation matrix would otherwise have to be supplied by domain experts.

Outline. Section 2 starts with reviewing existing work that our approaches builds upon. Section 3 gives our first contribution and show how to generalize semi-supervised learning to heterophily. Section 4 gives our second contribution and shows how heterophily can be learned from partially labeled data. Section 5 gives experiments, Section 6 reviews related work, before Section 7 concludes.

2. BACKGROUND

Here we describe three highly related bodies of work that the methods described in this paper build upon.

2.1 Semi-Supervised Learning (SSL)

Semi-supervised learning methods (SSL) derive their formalism usually by motivating a loss function consisting of (i) a fit term to existing labels, e.g. $(f_i - x_i)^2$ where x_i denotes the given label and f_i the learned label, and (ii) a smoothness term or regularizer, e.g. $(f_i - f_j)^2$. We focus our discussion here only on the most important aspects of SSL and refer to [26] and [4] for two excellent surveys on SSL. We follow here the exposition of [1] with two notable restrictions: (i) we assume a symmetric graph structure $\mathbf{W} = \mathbf{W}^\top$; (ii) we allow relabeling of already labeled information ($(f_i - x_i)^2 \neq 0$ possible). We focus on three methods, in particular:

1. The harmonic function method (HF) [27] minimizes the loss function $E(\mathbf{f}) = \sum_i (f_i - x_i)^2 + \frac{\mu}{2} \sum_{i,j} W_{ij} (f_i - f_j)^2$. Figure 3 shows $E(\mathbf{f})$ by using the Laplacian matrix: $\mathbf{L} = \mathbf{D} - \mathbf{W}$.
2. The linear neighborhood propagation (LNP) [22] is a variation in which the weights of the neighbors of a node i need to sum up to 1: $\sum_j W_{ij} = 1$. The update equation can thus be written slightly simpler by using $\alpha = \frac{\mu}{1+\mu}$.
3. The local and global consistency method (LGC) [25] is similar to HF except for normalizing the labeling function by the square root of the degrees of each node: $E(\mathbf{f}) = \sum_i (f_i - x_i)^2 + \frac{\mu}{2} \sum_{i,j} W_{ij} \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2$. The energy function can thus be written by using the normalized Laplacian matrix: $\mathbf{L}_n = \mathbf{D}^{-\frac{1}{2}} \mathbf{L} \mathbf{D}^{-\frac{1}{2}}$, and the update equations by using $\mathbf{L}_* = \mathbf{I} - \mathbf{L}_n$.

Multi-class classification with homophily. It is easy to extend existing label propagation algorithms to multi-class classification problems [22] by assigning with a vector to each node, where each entry represents the belief of a node in a particular labeling class. Suppose there are k classes, and the label set becomes $\mathcal{L} = \{1, 2, \dots, k\}$. Let \mathcal{M} be a set of $n \times k$ matrices with nonnegative real-valued entries. Any matrix $\mathbf{F} = [\mathbf{F}_1^\top, \mathbf{F}_2^\top, \dots, \mathbf{F}_n^\top]^\top \in \mathcal{M}$ corresponds to a specific classification on X that labels i as $y_i = \arg\max_{i \leq k} \mathbf{F}_{ij}$. Thus, \mathbf{F} can be viewed as the $n \times k$ -dimensional label matrix or as a function that assign labels for each data point. Initially, we set $\mathbf{F}_0 = \mathbf{T}$, where $\mathbf{T}_{ij} = 1$ if \mathbf{x}_i is labeled as j , and $\mathbf{T}_{ij} = 0$ otherwise, and for unlabeled points, $\mathbf{T}_{uj} = 0$ ($1 \leq j \leq k$).

³A Markov process is described by a state transition matrix \mathbf{T} where $T_{i,j} = \mathbb{P}[X^{(t+1)} = i | X^{(t)} = j]$ is the probability to end up in state i if starting at j in the previous step. In that formalism, \mathbf{T} is column stochastic and the column-wise sums are equal to 1: $\sum_i T_{i,j} = 1$.

	Update equation	Closed form	Loss function
Previous methods for binary case:			
HF [27]	$\mathbf{f} \leftarrow (\mathbf{I} + \mu\mathbf{D})^{-1}(\mathbf{x} + \mu\mathbf{W}\mathbf{f})$	$\mathbf{f} = (\mathbf{I} + \mu\mathbf{L})^{-1}\mathbf{x}$	$E(\mathbf{f}) = \ \mathbf{f} - \mathbf{x}\ ^2 + \mu\mathbf{f}^\top \mathbf{L}\mathbf{f}$
LNP [22]	$\mathbf{f} \leftarrow \bar{\alpha}\mathbf{x} + \alpha\mathbf{W}\mathbf{f}$	$\mathbf{f} = \bar{\alpha}(\mathbf{I} - \alpha\mathbf{W})^{-1}\mathbf{x}$	" " " " " "
LGC [25]	$\mathbf{f} \leftarrow \bar{\alpha}\mathbf{x} + \alpha\mathbf{L}_*\mathbf{f}$	$\mathbf{f} = \bar{\alpha}(\mathbf{I} - \alpha\mathbf{L}_*)^{-1}\mathbf{x}$	$E(\mathbf{f}) = \ \mathbf{f} - \mathbf{x}\ ^2 + \mu\mathbf{f}^\top \mathbf{L}_n\mathbf{f}$
FaBP [10]	$\mathbf{f} \leftarrow \mathbf{x} + 2h\mathbf{W}\mathbf{f}$	$\mathbf{f} = (\mathbf{I} - 2h\mathbf{W})^{-1}\mathbf{x}$	$E(\mathbf{f}) = \ \mathbf{f} - \mathbf{x} - 2h\mathbf{W}\mathbf{f}\ ^2$ (shown in this paper)
Previous methods for multi-class case:			
HF	$\mathbf{F} \leftarrow (\mathbf{I} + \mu\mathbf{D})^{-1}(\mathbf{X} + \mu\mathbf{W}\mathbf{F})$	$\mathbf{F} = (\mathbf{I} + \mu\mathbf{L})^{-1}\mathbf{X}$	$E(\mathbf{F}) = \ \mathbf{F} - \mathbf{X}\ ^2 + \frac{\mu}{2} \sum_{i,j} W_{ij} \sum_k (F_{ik} - F_{jk})^2$
LNP	$\mathbf{F} \leftarrow \bar{\alpha}\mathbf{X} + \alpha\mathbf{W}\mathbf{F}$	$\mathbf{F} = \bar{\alpha}(\mathbf{I} - \alpha\mathbf{W})^{-1}\mathbf{X}$	" " " " " "
LGC	$\mathbf{F} \leftarrow \bar{\alpha}\mathbf{X} + \alpha\mathbf{L}_*\mathbf{F}$	$\mathbf{F} = \bar{\alpha}(\mathbf{I} - \alpha\mathbf{L}_*)^{-1}\mathbf{X}$	$E(\mathbf{F}) = \ \mathbf{F} - \mathbf{X}\ ^2 + \frac{\mu}{2} \sum_{i,j} W_{ij} \sum_k (\frac{F_{ik}}{\sqrt{d_i}} - \frac{F_{jk}}{\sqrt{d_j}})^2$
LinBP [6]	$\mathbf{F} \leftarrow \mathbf{X} + \mathbf{W}\mathbf{F}\mathbf{H}$	$\text{vec}(\mathbf{F}) = (\mathbf{I} - \mathbf{H} \otimes \mathbf{W})^{-1}\text{vec}(\mathbf{X})$	$E(\mathbf{F}) = \ \mathbf{F} - \mathbf{X} - \mathbf{W}\mathbf{F}\mathbf{H}\ ^2$ (shown in this paper)

Figure 3: HF: harmonic function method (according to [1]), LNP: linear neighborhood propagation, LGC: local and global consistency method, FaBP: fast belief propagation. LinBP: linearized belief propagation. The symbol " is used for repeated entries

Figure 3 show the required changes to the loss function, update equations and closed-form solutions for HF, LNP and LGC, respectively (we will discuss FaBP and LinBP separately). Importantly, notice that labels for different classes do not interact with each other. In other words, they have no influence on each other.

2.2 Locally Linear Embedding (LLE)

Locally linear embedding (LLE) [15] is a method to derive compact representations of high-dimensional data by building a linear relationship among neighboring points. It is originally an unsupervised learning algorithm. We will later use the formulation for our SSL scenario, namely estimating the heterophily matrix \mathbf{H} instead of the embedding \mathbf{W} from data. We follow here exactly the exposition in the original paper [11] while using our notation.

The LLE algorithm assumes the data consist of n real-valued vectors \mathbf{x}_i , each of dimensionality k . LLE then reconstructs each data point from its neighbors by constructing a locally linear combination. Reconstruction errors are measured by the loss function:

$$E(\mathbf{W}) = \sum_i \|\mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j\|^2 \quad (1)$$

which adds up the squared distances between all the data points and their reconstructions. The weights W_{ij} summarize the contribution of the j -th data point to the i -th reconstruction. To compute the weights W_{ij} , one minimizes the cost function subject to the constraint of all rows of the weight matrix summing to one: $\sum_j W_{ij} = 1$. The optimal weights W_{ij} are found by solving a least-squares problem and can be computed in closed form.

Consider a particular data point \mathbf{x} with neighbors $\boldsymbol{\eta}_j$ and sum-to-one reconstruction weights \mathbf{w} . The reconstruction error $\|\mathbf{x} - \sum_{j=1}^K w_j \boldsymbol{\eta}_j\|^2$ is minimized in three steps: First, evaluate inner products between neighbors to compute the neighborhood correlation matrix, $C_{jk} = \boldsymbol{\eta}_j^\top \boldsymbol{\eta}_k$ and its matrix inverse, \mathbf{C}^{-1} . Second, compute the Lagrange multiplier, $\lambda = \alpha/\beta$, that enforces the sum-to-one constraint, where $\alpha = 1 - \sum_{jk} C_{jk}^{-1}(\mathbf{x}\boldsymbol{\eta}_k)$ and $\beta = \sum_{jk} C_{jk}^{-1}$. Third, compute the reconstruction weights: $w_j = \sum_k C_{jk}^{-1}(\mathbf{x}\boldsymbol{\eta}_k + \lambda)$

In the final step of LLE, each high-dimensional observation \mathbf{x}_i is mapped to a low-dimensional vector \mathbf{f}_i representing global internal coordinates on the manifold. This is done by choosing $d < k$ -dimensional coordinates \mathbf{f}_i to minimize

the embedding cost function

$$E(\mathbf{f}) = \sum_i \|\mathbf{f}_i - \sum_j W_{ij}\mathbf{f}_j\|^2 \quad (2)$$

This cost function, like the previous one, is based on locally linear reconstruction errors, but here we fix the weights W_{ij} while optimizing the coordinates \mathbf{f}_i .

2.3 Linearized Belief Propagation (LinBP)

Another widely used methods for semi-supervised reasoning in networked data is *Belief Propagation* (BP) [18]. Similar to the other SSL methods, BP helps to propagate the information from a few labeled nodes throughout the network by iteratively propagating information between neighboring nodes. In addition, it can handle the case of multiple classes influencing each other. For graphs with loops, however, BP has well-known convergence problems (see [18] for a detailed discussion from a practitioner's point of view). While there is a lot of work on convergence of BP [5, 12], *exact* criteria for convergence are not known [13, Sec. 22], and practical use of BP is still non-trivial [18].

Motivated by this, Koutra et al. [10] proposed to linearize belief propagation for the case of two classes and proposed fast belief propagation (FaBP) as a method to propagate existing knowledge of homophily or heterophily to unlabeled data. This framework allows to specify a *homophily factor* h ($h > 0$ for homophily or $h < 0$ for heterophily) and to then use this algorithm with exact convergence criteria for binary classification (e.g., yes/no or male/female).

In [6], we have recently solved the problem for the multi-class case and proposed Linearized Belief Propagation (LinBP) as an efficient linearization of belief propagation on pairwise Markov random fields.⁴ Our observations stem from the insight that the original update equations of belief propagation (here written compactly in matrix notation by using the symbol \odot for the Hadamard product⁵)

$$\begin{aligned} \mathbf{f}_i &\propto \mathbf{x}_i \odot \bigcirc_{j \in N(i)} \mathbf{m}_{ji} \\ \mathbf{m}_{it} &\propto \mathbf{H}(\mathbf{x}_i \odot \bigcirc_{j \in N(i) \setminus t} \mathbf{m}_{ji}) \end{aligned}$$

⁴Notice that in [6], we distinguished between two versions of linearized belief propagation: LinBP and LinBP*. The formulation of LinBP that we consider in this paper corresponds to LinBP* while we do not discuss the original LinBP version.

⁵The Hadamard product (also called component-wise multiplication operator), is defined by: $\mathbf{Z} = \mathbf{X} \odot \mathbf{Y} \Leftrightarrow Z(i, j) = X(i, j) \cdot Y(i, j)$.

can be approximated by linearized equations

$$\hat{\mathbf{f}}_i \leftarrow \hat{\mathbf{x}}_i + \frac{1}{k} \cdot \sum_{j \in N(i)} \hat{\mathbf{m}}_{ji} \quad (3)$$

$$\hat{\mathbf{m}}_{it} \leftarrow \hat{\mathbf{H}} \left(\hat{\mathbf{f}}_i - \frac{1}{k} \hat{\mathbf{m}}_{ti} \right) \quad (4)$$

by “centering” all variables around appropriate values. We call a vector or matrix \mathbf{x} “centered around c ” if all its entries are close to c and their average is exactly c . If a vector \mathbf{x} is centered around c , then the residual vector around c is defined as $\hat{\mathbf{x}} = [x_1 - c, x_2 - c, \dots]$. Accordingly, we denote a matrix $\hat{\mathbf{X}}$ as a residual matrix if each column and row vector corresponds to a residual vector.

Concretely, we centered the k -dimensional message vectors \mathbf{m} around 1, and centered all the other k -dimensional vectors/matrices around $1/k$. Thus, they become probability vectors and their entries have to sum up to 1. As a consequence, $\hat{\mathbf{H}} \in \mathbb{R}^{k \times k}$ is the *residual coupling matrix* that makes explicit the relative attraction and repulsion: the sign of \hat{H}_{ji} tells us if the class j attracts or repels class i in a neighbor, and the magnitude of \hat{H}_{ji} indicates the strength. Subsequently, this centering allows us to rewrite belief propagation in terms of the residuals. Importantly, the derived messages remain centered for any iteration and thus no normalization is necessary.

We have also given an iterative calculation of the final beliefs. Starting with an arbitrary initialization of $\hat{\mathbf{F}}$ (e.g., all values zero), we repeatedly compute the right hand side of the equations and update the values of $\hat{\mathbf{F}}$ until the process converges: We have shown that the solution for LinBP can be calculated by applying the following iterative update equation:

$$\hat{\mathbf{F}} \leftarrow \hat{\mathbf{X}} + \mathbf{W} \hat{\mathbf{F}} \hat{\mathbf{H}} \quad (\text{LinBP}) \quad (5)$$

Thus, the final beliefs of each node can be computed via elegant matrix operations and optimized solvers. In addition, this formalism allows a closed form solution:

PROPOSITION 1 (CLOSED-FORM [6]). *The closed-form solution for Eq. 5 is given by:*

$$\text{vec}(\hat{\mathbf{F}}) = (\mathbf{I} - \hat{\mathbf{H}} \otimes \mathbf{W})^{-1} \text{vec}(\hat{\mathbf{X}}) \quad (6)$$

Furthermore, by analyzing Eq. 6, we could derive exact (sufficient and necessary) conditions for convergence of our update equations based on the spectral radius (ρ) of the adjacency matrix \mathbf{W} and the affinity matrix \mathbf{H} :

LEMMA 2 (CONVERGENCE [6]). *Necessary and sufficient criterion for convergence of LinBP is:*

$$\text{LinBP converges} \Leftrightarrow \rho(\hat{\mathbf{H}}) < \frac{1}{\rho(\mathbf{W})} \quad (7)$$

In practice, we use the matrix \mathbf{H} as a scaled version of an original unscaled but centered matrix: $\hat{\mathbf{H}} = \epsilon_h \hat{\mathbf{H}}_0$.

3. GENERALIZING SSL FOR HETEROPHILY

We illustrate here generalizing the update equations for the harmonic functions (HF) method [27] to Harmonic functions with heterophily (HFH).

Derivation from regularization framework. In the following, we partially follow the exposition by Bengio et al. [1]. Assume n nodes with ℓ labeled nodes, and WLOG

all labeled nodes have index $i \leq \ell$. The harmonic function method (HF) [27] then minimizes the loss function

$$E(\mathbf{f}) = \sum_i^\ell (f_i - x_i)^2 + \frac{\mu}{2} \sum_{i,j}^n W_{ij} (f_i - f_j)^2$$

Let \mathbf{G} be the diagonal matrix with $G_{i,i} = 1$ if $i \leq \ell$, and $G_{i,i} = 0$, otherwise and write $\mathbf{L} = \mathbf{D} - \mathbf{W}$ for the graph Laplacian matrix. Thus, we can alternatively write:

$$E(\mathbf{f}) = \|\mathbf{G}\mathbf{f} - \mathbf{G}\mathbf{x}\|^2 + \mu \mathbf{f}^\top \mathbf{L} \mathbf{f}$$

We get as derivative:

$$\frac{\partial E(\mathbf{f})}{\partial \mathbf{f}} = \mathbf{G}(\mathbf{f} - \mathbf{x}) + \mu \mathbf{L} \mathbf{f}$$

Setting the derivative to 0, we get:

$$\begin{aligned} (\mathbf{G} + \mu \mathbf{L}) \mathbf{f} &= \mathbf{G} \mathbf{x} \\ \mathbf{f} &= (\mathbf{G} + \mu \mathbf{L})^{-1} \mathbf{G} \mathbf{x} \end{aligned}$$

Modulating the existing update equations. Intuitively, the previous equations can be seen as a weighted average of the neighbors’ current labels, where for labeled examples we overwrite the propagated values with the initial label.

In the following, we write slightly different update equations. Let’s call $\mathbf{P} := \mathbf{D}^{-1} \mathbf{W}$ the row-stochastic adjacency matrix and write the update equation as follows:

$$\mathbf{f}' \leftarrow \mathbf{P} \mathbf{f}$$

However, we always use the explicit label if available, i.e. the labels of explicit beliefs constrained to be equal \mathbf{x} :

$$f_i \leftarrow \begin{cases} f'_i & \text{if } i \text{ implicit node} \\ x_i & \text{if } i \text{ explicit node} \end{cases}$$

Another way to write this is by defining $\mathbf{P}^\mathbf{x}$ with:

$$P_{i,j}^\mathbf{x} \leftarrow \begin{cases} P_{i,j} & \text{if } i \text{ implicit node} \\ 0 & \text{if } i \text{ explicit node} \end{cases}$$

and writing:

$$\mathbf{f} \leftarrow \mathbf{x} + \mathbf{P}^\mathbf{x} \mathbf{f}$$

In other words, we define a new matrix that is the result of deleting all column for the indices of explicit beliefs. An alternative way to write this is by defining a column vector \mathbf{e} that has an entry 1 for all indices of explicit beliefs, and 0 otherwise. Then, $\bar{\mathbf{e}} = \mathbf{1} - \mathbf{e}$, and $\mathbf{P}^\mathbf{x} = \mathbf{1} \cdot \bar{\mathbf{e}}^\top \odot \mathbf{P}$.

New assume we have k different classes, and we propagate each separately. However, the nodes are the same across all classes. Then the multi-class version is

$$\mathbf{F} \leftarrow \mathbf{X} + \mathbf{P}^\mathbf{x} \mathbf{F}$$

The, we get the closed form after convergence:

$$\begin{aligned} \mathbf{F} &= \mathbf{X} + \mathbf{P}^\mathbf{x} \mathbf{F} \\ \mathbf{F} &= (\mathbf{I} - \mathbf{P}^\mathbf{x})^{-1} \mathbf{X} \end{aligned}$$

New, we can next introduce “heterophily propagation,” i.e. we modulate the vectors before propagating them:

$$\mathbf{F} \leftarrow \mathbf{X} + \mathbf{P}^\mathbf{x} \mathbf{F} \mathbf{H} \quad (\text{HHF}) \quad (8)$$

We can now use the exact same mathematics we had derived in [6] in order to determine the closed-form as follows:

$$\text{vec}(\mathbf{F}) = (\mathbf{I} - \mathbf{H} \otimes \mathbf{P}^x)^{-1} \text{vec}(\mathbf{X})$$

Notice that the above update equations will always converge as long as the spectral radius of $\mathbf{P}^x < 1$. This is the case if each connected component has at least one explicit belief

Constrained LinBP (LinBP^x). We propose here a “constraint” variant of LinBP, where the labels of explicit nodes (sometimes depicted as \mathbf{f}_i) are constrained to the explicit beliefs (\mathbf{x}_i). We use the same trick as before by defining \mathbf{W}^x with:

$$W_{i,j}^x \leftarrow \begin{cases} W_{i,j} & \text{if } i > l \\ 0 & \text{if } i \leq l \end{cases}$$

to get:

$$\begin{aligned} \hat{\mathbf{F}} &\leftarrow \hat{\mathbf{X}} + \mathbf{W}^x \hat{\mathbf{F}} \hat{\mathbf{H}} \\ \text{vec}(\hat{\mathbf{F}}) &= (\mathbf{I} - \hat{\mathbf{H}} \otimes \mathbf{W}^x)^{-1} \text{vec}(\hat{\mathbf{X}}) \end{aligned}$$

We call the resulting formulation LinBP^x, where the x notation means the constrained (or “fixed”) version of some propagation matrix.

Generalization. More generally, we propose here a general form of update equations and closed-form where the matrix \mathbf{M} can be chosen from existing semi-supervised learning methods.

$$\begin{aligned} \mathbf{F} &\leftarrow \mathbf{X} + \mathbf{M} \mathbf{F} \mathbf{H} \\ \text{vec}(\mathbf{F}) &= (\mathbf{I} - \mathbf{H} \otimes \mathbf{M})^{-1} \text{vec}(\mathbf{X}) \end{aligned}$$

We have seen three examples: \mathbf{W} for LinBP, \mathbf{P}^x for HFH^x and \mathbf{W}^x for LinBP^x.

4. LEARNING HETEROPHILY FROM DATA

In this section, we introduce our framework for learning heterophily from partially labeled data.

4.1 Baseline approach: MHE

We first introduce a straight-forward baseline method that we call *Myopic Heterophily Estimation* (MHE). We call it “myopic” as it tries to infer the relative frequencies between different classes in the network by a straightforward frequency calculation, followed by a transformation into a symmetric doubly-stochastic matrix.

Given partially labeled $n \times k$ -matrix \mathbf{X}' , with $X'(i, c) = 1$, if node i has label c . Recall that some nodes have no label. Then the Matrix $n \times k$ -matrix $\mathbf{N} := \mathbf{W} \mathbf{X}'$ contains the number of neighbors of a certain class for each node, i.e. $N(i, c)$ determines the number of neighbors from i that are of class c . Next, the matrix $\mathbf{H}' = \mathbf{X}'^T \mathbf{N} = \mathbf{X}'^T \mathbf{W} \mathbf{X}$ has as entry $H(c, d)$ the number of nodes of class d that are neighbors of nodes of class c . This matrix is symmetric, but not stochastic. One can make it row-stochastic by dividing each row by the sum of each row. Let’s call this matrix \mathbf{H}'' . This matrix, however, is not symmetric anymore.

What are proposing as baseline method is to use the symmetric, doubly stochastic matrix \mathbf{H}^* (i.e., it fulfills the conditions $\mathbf{H} = \mathbf{H}^T$ and $\mathbf{H} \mathbf{1} = \mathbf{1}$) that is closest to the matrix

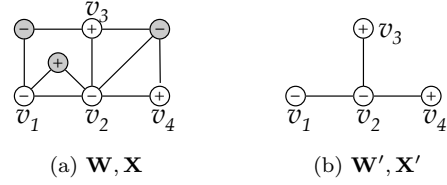


Figure 4

\mathbf{H}'' by using the Frobenius norm as distance:

$$\begin{aligned} \underset{\mathbf{H}}{\text{argmin}} \quad & \|\mathbf{H} - \mathbf{H}''\| \quad (\text{MHE}) \\ \text{subject to} \quad & \mathbf{H} = \mathbf{H}^T \\ & \mathbf{H} \mathbf{1} = \mathbf{1} \end{aligned}$$

This problem can be solved efficiently with Algorithm 1 in [24] that finds a Frobenius-norm optimum doubly stochastic approximation to a given matrix

Notice that in the case of an incompletely labeled graph, it suffices to consider only the subgraph induced by the labeled nodes.

EXAMPLE 3. Consider the graph in Fig. 4a. Gray nodes are showing unlabeled nodes. Thus the graph we need to consider is only Fig. 4b. + and - show classes 1 and 2, respectively. Then $\mathbf{H}' = \begin{bmatrix} 0 & 2 \\ 2 & 2 \end{bmatrix}$. And $\mathbf{H}'' = \begin{bmatrix} 0 & 1 \\ 1 & 2 \end{bmatrix}$. MHE leads to $\mathbf{H}^* = \begin{bmatrix} \frac{1}{4} & \frac{3}{4} \\ \frac{3}{4} & \frac{1}{4} \end{bmatrix}$. In contrast, LHE leads to $\mathbf{H} = \begin{bmatrix} \frac{1}{3} & \frac{2}{3} \\ \frac{2}{3} & \frac{1}{3} \end{bmatrix}$.

4.2 Improved approach: LHE

We next give a simple energy minimization framework for LinBP. This formalization will allow us in a latter step to solve the problem of deriving the optimal heterophily matrix.

PROPOSITION 4 (LINBP LOSS FUNCTION). The energy function minimized by LinBP is

$$E(\mathbf{F}) = \|\mathbf{F} - \mathbf{X} - \mathbf{W} \mathbf{F} \mathbf{H}\|^2 \quad (9)$$

The proof follows immediately from our proof of convergence of LinBP [6].

We would next like to compare the loss functions for LLE (Eq. 1), LinBP (Eq. 9), and HF (Fig. 3):

$$E(\mathbf{W}) = \sum_i \|\mathbf{x}_i - \sum_j W_{ij} \mathbf{x}_j\|^2 \quad (\text{LLE})$$

$$E(\mathbf{F}) = \|\mathbf{F} - \mathbf{X} - \mathbf{W} \mathbf{F} \mathbf{H}\|^2 \quad (\text{LinBP})$$

$$E(\mathbf{F}) = \|\mathbf{F} - \mathbf{X}\|^2 + \frac{\mu}{2} \sum_{i,j} W_{ij} \sum_k (F_{ik} - F_{jk})^2 \quad (\text{HF})$$

Notice several similarities and differences: (i) LLE tries to learn \mathbf{W} , whereas LinBP and HF try to learn \mathbf{F} . (ii) By ignoring explicit labels \mathbf{X} (i.e., ignoring the fit term in HF and ignoring the vector in LinBP), and replacing \mathbf{H} with the identity matrix \mathbf{I} , all three equations become the same. We thus propose the following two formalizations:

1. First, we propose to estimate the heterophily matrix \mathbf{H} from partially labeled data via the following loss function

$$E(\mathbf{H}) = \|\mathbf{X} - \mathbf{W}\mathbf{X}\mathbf{H}\|^2 \quad (\text{LHE}) \quad (10)$$

This is justified for three reasons: (i) The difference between \mathbf{X} and \mathbf{F} is not important for learning of the heterophily matrix. These are two different loss functions and we can focus only on one of them; (ii) We can use the mathematical machinery developed for LLE for our own problem setup; and (iii) This formalism explains heterophily for both LinBP and HF, as we show next.

2. Second, we propose to generalize the regularization term of HF (and analogously for LNP and LGC) as follows:

$$R(\mathbf{F}, \mathbf{H}) = \frac{1}{2} \sum_{i,j} W_{ij} \|\mathbf{F}_i - \mathbf{F}_j \mathbf{H}\|^2 \quad (\text{HHF}) \quad (11)$$

Here, we write \mathbf{F}_i as short notation for the i -th row vector of the label matrix \mathbf{F} . This is justified for two reasons: (i) For $\mathbf{H} = \mathbf{I}$, we get back the original formulation of HF.⁶ (ii) As long as each entry \mathbf{F}_i is stochastic, the equations make sure that all updates and final labeling functions remain stochastic.

Thus, we propose in this section to learn the heterophily matrix via the following simple convex optimization problem:

$$\begin{aligned} & \underset{\mathbf{H}}{\operatorname{argmin}} \quad \|\mathbf{X} - \mathbf{W}\mathbf{X}\mathbf{H}\| \quad (\text{LHE}) \quad (12) \\ & \text{subject to} \quad \mathbf{H} = \mathbf{H}^\top \\ & \quad \quad \quad \mathbf{H} \mathbf{1} = \mathbf{1} \end{aligned}$$

Notice that we can transform the simple constraint optimization problem from Eq. 12 into an even simple unconstrained optimization problem. A $k \times k$ -dimensional doubly stochastic matrix $\frac{k(k-1)}{2}$ degrees of freedom, thus $\frac{k(k+1)}{2}$ constraints which we need to impose as boundary conditions. In other words, we can pose the optimization problem simply over the degrees of freedom of the matrix. For example, for a $k = 3$, we have 3 degrees of freedom:

$$\mathbf{H} = \begin{bmatrix} h_1 & h_2 & 1-h_1-h_2 \\ h_2 & h_3 & 1-h_2-h_3 \\ 1-h_1-h_2 & 1-h_2-h_3 & h_1+2h_2+h_3-1 \end{bmatrix}$$

Thus we have an unconstrained optimization problem

$$\underset{\mathbf{h}}{\operatorname{argmin}} \quad \|\mathbf{X} - \mathbf{W}\mathbf{X}\mathbf{H}(\mathbf{h})\| \quad (\text{LHE}) \quad (13)$$

We found that the latter formulation is faster in practice.

5. EXPERIMENTS

We evaluate our method with regard to two questions: (1) How well does our Linear Heterophily Estimation (LHE) work as compared to the baseline method? (2) How well do our combined methods scale with the size of the network?

5.1 Quality

We chose to evaluate our technique on carefully controlled synthetic data only, as this allows us to change the ground truth and evaluate the accuracy of our techniques as result of

⁶HHF stands for Heterophily Harmonic Functions.

systematic changes to problem parameters. We repeated the experiments with various synthetic data sets, and will focus on one particular choice of parameters to illustrate the main conclusions. Figure 5h summarizes our overall experimental setup.

Data generation: $\mathbf{W}, \mathbf{X}, \mathbf{X}'$. We assume $k = 3$ classes and use the heterophily matrix from the introduction: $\mathbf{H} = \begin{bmatrix} 0.1 & 0.8 & 0.1 \\ 0.8 & 0.1 & 0.1 \\ 0.1 & 0.1 & 0.8 \end{bmatrix}$. We create a graph with $n = 100$ nodes and assign each node one of the 3 classes (we assumed equal prior probabilities of a node being a particular class: $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$). This leads to our $n \times k$ indicator ground truth labeling matrix \mathbf{X} . We then add for each node exactly $\frac{m}{2} \in \{1, 2, \dots, 10\}$ number of undirected edges to other nodes (leading to m average degree). The random choice of a neighbor reflects the homophily matrix \mathbf{H} as follows: We first randomly choose the type of neighbor to connect to (i.e., a node i of class c_1 is connected to another node j of class c_2 with probability $H(c_1, c_2)$), and then randomly sample a node of that class. This leads to the binary, symmetric adjacency matrix \mathbf{W} . We then remove a random fraction $p \in \{0.1, \dots, 0.95\}$ of the labels from \mathbf{X} , leading to the partially labeled data \mathbf{X}' . We vary the parameters and repeat this experiment.

Homophily estimation: $\mathbf{H}_2, \mathbf{H}_1$. We use Eq. 12 to estimate \mathbf{H}_2 on the partially labeled graph with edges \mathbf{W} and labels \mathbf{X}' . We calculate the Frobenius norm $\|\mathbf{H} - \mathbf{H}_2\|$ as our estimation error. We also estimate \mathbf{H}_1 on the completely labeled graph with \mathbf{X} . This allows us to later analyze and interpret the relative contribution of (i) more accurate estimation of \mathbf{H} and (ii) larger fraction of labeled information, to the labeling accuracy.

Label propagation: \mathbf{F}' . We transform our estimated homophily matrices $\mathbf{H}_2, \mathbf{H}_1$, and the original matrix \mathbf{H} into a centered matrix around 0: $\hat{\mathbf{H}} = \epsilon_H (\mathbf{H} - [\frac{1}{k}]_k)$, whereby systematically varying ϵ_H . We then use Eq. 5 to propagate the labeling information from \mathbf{X}' throughout the graph.

Quality assessment. For each node in the hold out set, we calculate the label with maximum belief in \mathbf{F}' and then evaluate labeling accuracy as the percentage of correctly retrieved labels on the hold-out set only. Notice that accuracy of $\frac{1}{3}$ corresponds to random assignment of labels.

Discussion. Overall, our method works well. For example, for $m = 2$ (i.e., average degree of a node is 2) and deleting 50% of the labels ($p = 0.5$), we have an average labeling accuracy of 0.76 for $\epsilon_H = 0.01$ (Fig. 5a). This is surprising, given that each column of the homophily matrix itself has relative high entropy $H([0.8, 0.1, 0.1]) \approx 0.92$ (as compared to the random vector $H([\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]) \approx 1.58$). Increasing the connectivity of the graph, also increases the labeling accuracy (more each unlabeled node, more edges will lead to labeled neighbors). However, it does not increase the estimation accuracy for \mathbf{H} : The average estimation error $\|\mathbf{H} - \mathbf{H}_2\|$ increases with the fraction p of removed labels, but is independent of the number of edges per node m (Fig. 5e and Fig. 5f).

The impact of the choice of the scaling factor ϵ_H for the label propagation homophily matrix $\hat{\mathbf{H}}$ is interesting. Below a certain threshold, accuracy is constant (e.g., 0.84 for $\epsilon_H = 0.01$), but then increases slightly up to the point of divergence (e.g., 0.86 for $\epsilon_H = 0.3$ in Fig. 5g for $m = 4, p = 0.5$). This result is consistent across different choices of parameters. For example, in the case of $m = 4$, the average boundary between convergence and divergence (Eq. 7)

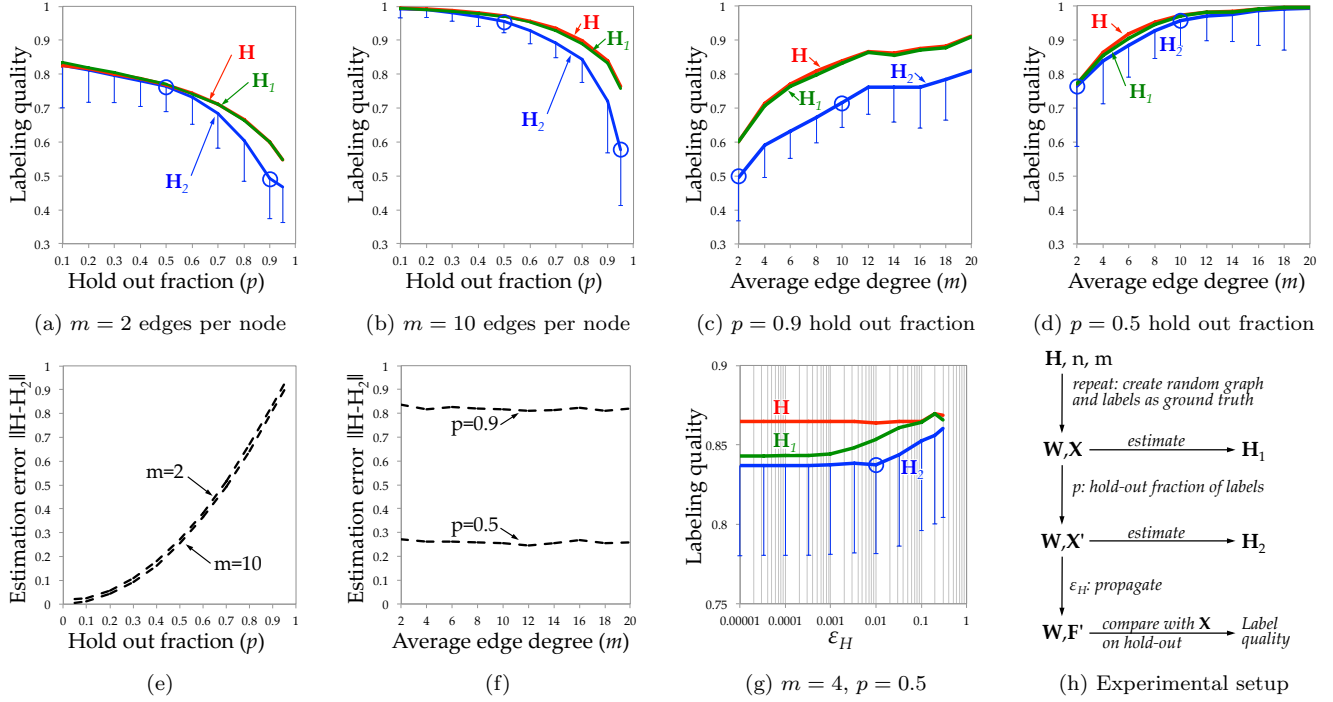


Figure 5: Results from experiments on graph with $n = 100$ nodes. Data points with blue circles are referenced in the text.

is $\epsilon_{\max} \approx 0.315$ (the graph in Fig. 5g ends at $\epsilon_h = 0.3$). This suggests that higher choices of ϵ_H (meaning the information is propagated further through the graph) allow to partially compensate for missing labels. For example, an unlabeled node may have no labeled neighbor, but labeling information is still passed through those neighbor nodes. All figures, except Fig. 5g, are drawn using $\epsilon_h = 0.01$.

Comparison baseline. We also compared our approach of estimating the matrix with LHE against MHE. Figure 6a shows the result for a graph with $n = 100$ nodes and $m = 2$ edges per node as a function of the hold-out fraction. For example for $f = 0.9$ (i.e. only 10% of nodes are labeled) NHE stops working (we do not have enough neighbors of all classes to estimate a matrix), while LHE combined with LinBP still 55% accuracy of determining the correct labels of the remaining 90 nodes.

5.2 Scalability

For the scalability experiments, we implemented our learning framework Python 2.7. We use `fmin_slsqp` in `scipy.optimize` to minimize our function using Sequential Least Squares Programming. Figure 6b shows the results for graphs with either average $m = 2$ or $m = 10$ edges per node. The time for LHE for 1M nodes and 10M edges is 31sec and LinBP around 7sec for 10 iterations.

6. RELATED WORK

Our work is related to multiple existing works on semi-supervised learning. We previously discussed in detail the harmonic function method (HF) [27], linear neighborhood propagation (LNP) [22], local and global consistency method (LGC) [25], locally linear embedding (LLE) [15], fast belief propagation (FaBP) [10], and Linearized Belief Propagation (LinBP) [6]. In [27] a symmetric weight matrix \mathbf{W} is learned

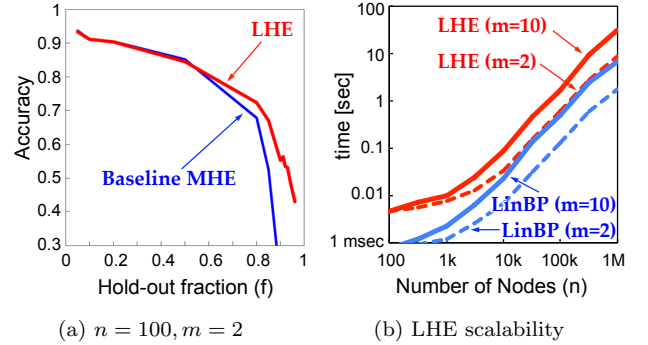


Figure 6: (a) Quality of our method of Linear Heterophily Estimation (LHE) as compared to the baseline method Myopic Heterophily Estimation (MHE) for increasingly larger hold-out fractions. (b) Scalability of our combined methods (LHE plus LinBP) to 1M nodes with either 2M or 10M edges.

but they assume simple homophily and do not account for more complex heterophily relations. In [22] a multi-class classification problems is considered. Though, each of the classes is treated separately and they never interact (mixing/modularization) with other classes. The works [7] and [21] integrate dissimilarity into the label learning process: one can indicate that two neighboring nodes should have different labels. Our proposed approach allows for much richer interaction between different classes. Also, [23] formulate a local learning regularizer (LL-Reg): learned to predict the label of each data point based on the neighbors and their labels.

To the best of our knowledge, the type of linear mixing described in this paper, by using a heterophily matrix \mathbf{H} , has

only been described before in our recent work on LinBP [6]. We are not aware of any linear learning framework for heterophily from existing data. We refer to [11], [26], [4], and [1] for excellent exposition and comparison of various methods.

7. CONCLUSIONS

In this paper, we proposed a novel semi-supervised learning formulation that relies on two novel components: First, it allows to use not only similarity and dissimilarity, but any type of mutual coupling strengths between different classes of nodes (we abstract this with the doubly stochastic heterophily matrix). Second, we showed how to estimate the heterophily matrix based on partially labeled data. The learned heterophily can subsequently be used to label the remaining data. We also showed how our framework generalizes a number of existing frameworks and naturally extends them from homophily to heterophily. Finally we provide experiments that illustrate the effectiveness of our method, plus detailed insights about the implications of parameters of the problem on our ability to correctly label data.

Acknowledgements. I like to thank Christos Faloutsos for stimulating discussions, helpful comments, and for getting me excited about Linear Algebra problems in the first place (thanks ☺ !). I also thank Stephan Günnemann for comments on an earlier version of this paper, and Jeff Schneider and Fernando de la Torre for helpful pointers.

8. REFERENCES

- [1] Y. Bengio, O. Delalleau, and N. L. Roux. Label propagation and quadratic criterion. In O. Chapelle, B. Schölkopf, and A. Zien, editors, *Semi-supervised learning*, pp. 193–216. MIT Press, 2006.
- [2] B. Bollobás, C. Borgs, J. Chayes, and O. Riordan. Directed scale-free graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA ’03, pp. 132–139, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.
- [3] L. Breiman. Statistical modeling: The two cultures. *Statistical Science*, 16(3):199–215, 2001.
- [4] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. MIT Press, Cambridge, Mass., 2006.
- [5] G. Elidan, I. McGraw, and D. Koller. Residual belief propagation: Informed scheduling for asynchronous message passing. In *UAI*, 2006.
- [6] W. Gatterbauer, S. Günnemann, D. Koutra, and C. Faloutsos. Linearized and single-pass belief propagation. *PVLDB*, 8(5), 2015. (full version: CoRR abs/1406.7288).
- [7] A. B. Goldberg, X. Zhu, and S. J. Wright. Dissimilarity in graph-based semi-supervised classification. In *AISTATS*, pp. 155–162, 2007.
- [8] A. Goldenberg, A. X. Zheng, S. E. Fienberg, and E. M. Airoldi. A survey of statistical network models. *Foundations and Trends in Machine Learning*, 2(2):129–233, 2009.
- [9] P. W. Holland, K. B. Laskey, and S. Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109 – 137, 1983.
- [10] D. Koutra, T.-Y. Ke, U. Kang, D. H. Chau, H.-K. K. Pao, and C. Faloutsos. Unifying guilt-by-association approaches: Theorems and fast algorithms. In *ECML/PKDD (2)*, pp. 245–260, 2011.
- [11] Q. Lu and L. Getoor. Link-based classification. In *ICML*, pp. 496–503, 2003.
- [12] J. M. Mooij and H. J. Kappen. Sufficient conditions for convergence of the sum-product algorithm. *IEEE Transactions on Information Theory*, 53(12):4422–4437, 2007.
- [13] K. P. Murphy. *Machine learning: a probabilistic perspective*. Adaptive computation and machine learning series. MIT Press, Cambridge, MA, 2012.
- [14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [15] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–6, Dec 2000.
- [16] M. H. Schneider and S. A. Zenios. A comparative study of algorithms for matrix balancing. *Oper. Res.*, 38(3):439–455, May 1990.
- [17] P. Sen and L. Getoor. Link-based classification. Technical report, University of Maryland Technical Report CS-TR-4858, February 2007.
- [18] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.
- [19] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 06 1964.
- [20] L. Stockmeyer. Planar 3-colorability is polynomial complete. *SIGACT News*, 5(3):19–25, July 1973.
- [21] W. Tong and R. Jin. Semi-supervised learning by mixed label propagation. In *AAAI*, pp. 651–656. AAAI Press, 2007.
- [22] F. Wang and C. Zhang. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2008.
- [23] M. Wu and B. Schölkopf. Transductive classification via local learning regularization. In *AISTATS*, pp. 628–635, 2007.
- [24] R. Zass and A. Shashua. Doubly stochastic normalization for spectral clustering. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pp. 1569–1576. MIT Press, 2006.
- [25] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2003.
- [26] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.
- [27] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pp. 912–919, 2003.

APPENDIX

A. NOMENCLATURE

n	number of nodes
k	number of classes
\mathbf{W}	$n \times n$ weighted symmetric adjacency matrix
\mathbf{P}	$n \times n$ row stochastic adjacency matrix
$\mathbf{W}^x, \mathbf{P}^x$	“fixed” variants: removed columns for explicit beliefs
x_i	given label of node i (if labeled)
f_i	inferred label of node i
\mathbf{X}	$n \times k$ given label distribution
\mathbf{F}	$n \times k$ inferred label distribution
\mathbf{L}	$n \times n$ Laplacian matrix
\mathbf{L}_n	$n \times n$ normalized Laplacian matrix
\mathbf{H}	$k \times k$ coupling matrix with $H_{i,j}$ indicating the influence of class i of a sender on class j of the recipient
$\hat{\mathbf{X}}, \hat{\mathbf{F}}, \hat{\mathbf{H}}$	residual matrices centered around $\frac{1}{k}$
$\hat{\mathbf{H}}_o$	unscaled, centered coupling matrix: $\hat{\mathbf{H}} = \epsilon_H \hat{\mathbf{H}}_o$
ϵ_H	scaling factor
\mathbf{I}_k	k -dimensional identity matrix
\mathbf{X}^\top	transpose of matrix \mathbf{X}
$\text{vec}(\mathbf{X})$	vectorization of matrix \mathbf{X}
$\mathbf{X} \otimes \mathbf{Y}$	Kronecker product between matrices \mathbf{X} and \mathbf{Y}
$\rho(\mathbf{X})$	spectral radius of a matrix \mathbf{X}
$\mathbf{X}_{i:}$	i -th row vector of \mathbf{X}